

Bounded Policy Iteration for Decentralized POMDPs

Daniel S. Bernstein
University of Massachusetts Amherst
Eric A. Hansen
Mississippi State University
Shlomo Zilberstein
University of Massachusetts Amherst

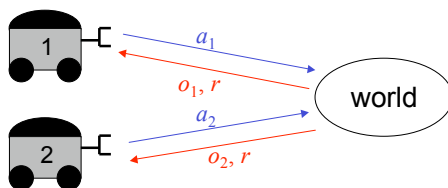
August 2, 2005

Team Decision Making



- How can we achieve intelligent coordination in spite of stochasticity and limited information?
- Application areas: networking, e-commerce, multi-robot systems, space exploration systems

Decentralized POMDP



- Multiple cooperating agents controlling a Markov process
- Each receives its own local observations

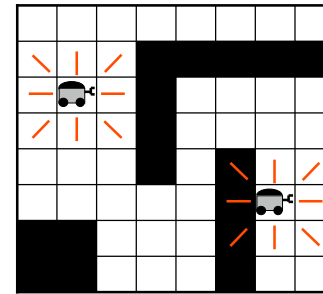
DEC-POMDP – Formal Definition

- A **DEC-POMDP** is $\langle S, A_1, A_2, P, R, \Omega_1, \Omega_2, O \rangle$, where
 - S is a finite state set, with initial state s_0
 - A_1, A_2 are finite action sets
 - $P(s, a_1, a_2, s')$ is a state transition function
 - $R(s, a_1, a_2)$ is a reward function
 - Ω_1, Ω_2 are finite observation sets
 - $O(s, o_1, o_2)$ is an observation function
- Straightforward generalization to n agents

DEC-POMDP – More Definitions

- A **local policy** is a mapping $\delta_i : \Omega_i^* \rightarrow A_i$
- A **joint policy** is a pair $\langle \delta_1, \delta_2 \rangle$
- Goal is to maximize expected discounted reward over an infinite horizon
- Although *execution* is distributed, *planning* can be centralized

An Illustrative Example



- States: grid cell pairs
- Actions: $\uparrow, \downarrow, \leftarrow, \rightarrow$
- Transitions: noisy
- Goal: meet quickly
- Observations: red lines

Optimal Algorithms

- Brute force search through policy space
 - Impractical for all but the tiniest problems
- Dynamic programming
 - Hansen, Bernstein & Zilberstein 04
 - Requires a large amount of memory
 - Currently can only solve very small problems

Bounded Policy Iteration

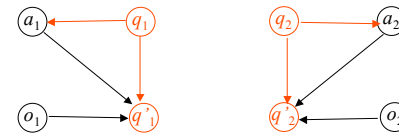
- **Memory:** fixed ahead of time
- **Time:** polynomial per iteration
- **Policy representation:** randomness and correlation used to offset memory limitations
- **Guarantee:** monotonic value improvement for *all* initial state distributions at *each* iteration
- Generalizes previous work on POMDPs
 - Poupart & Boutilier 03

Existing Approximation Algorithms

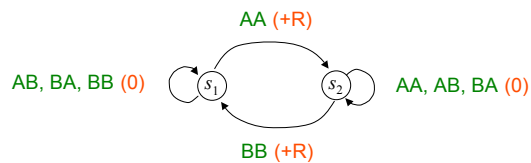
- Gradient ascent [Peshkin et al. 00](#)
 - Requires initial state distribution as input
- Joint Equilibrium-based Search [Nair et al. 03](#)
 - Doesn't address memory issues
- Bayesian game approach [Emery-Montemerlo et al. 04](#)
 - Requires initial state distribution as input

Independent Joint Controllers

- Can define a local controller for agent i to be a conditional distribution $P(a_i, q'_i | q_i, o_i)$
- An **independent joint controller** is $\prod_i P(a_i, q'_i | q_i, o_i)$



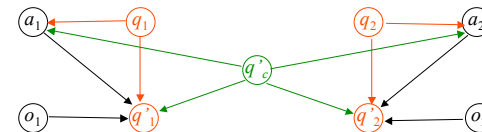
The Utility of Correlation



- Two agents, each with actions **A** and **B**
- Restricted to memoryless, open-loop policies
- Best policy is 1/2 **AA** and 1/2 **BB**

Correlated Joint Controllers

- A **correlation device** is a Markov chain $P(q'_c | q_c)$
- Controller is $\sum_{q'_c} P(q'_c | q_c) \prod_i P(a_i, q'_i | q_i, o_i, q'_c)$



- Random bits for the correlation device can be determined prior to execution time

Bounded Policy Iteration

- Start with an arbitrary joint controller
- Repeatedly apply **bounded backups** to improve the controller
- Bounded backups can be applied to local controller nodes or nodes of the correlation device

Bounded Backup for a Local Controller

- Choose a node q_i
- Try to find better parameters for q_i , assuming that the old parameters will be used from the second step onwards
- New parameters must yield value at least as high for all states and nodes of the other local controllers and correlation device

Bounded Backup for a Local Controller

Variables: $\varepsilon, P(a_i, q_i | q_i, o_i, q_c)$

Objective: Maximize ε

Constraints: $\forall s \in S, q_{-i} \in Q_{-i}, q_c \in Q_c$

$$V(s, \vec{q}, q_c) + \varepsilon \leq \sum_a P(\vec{a} | \vec{q}, q_c) \left[R(s, a) + \gamma \sum_{s', \vec{q}', q_c'} P(\vec{q}' | \vec{q}, \vec{a}, \vec{o}, q_c) P(s' | s, \vec{a}) P(q_c' | q_c) V(s', \vec{q}', q_c') \right]$$

Value Improvement Theorem

Theorem: *Performing a bounded backup produces a new joint controller with value at least as high for every initial state distribution.*

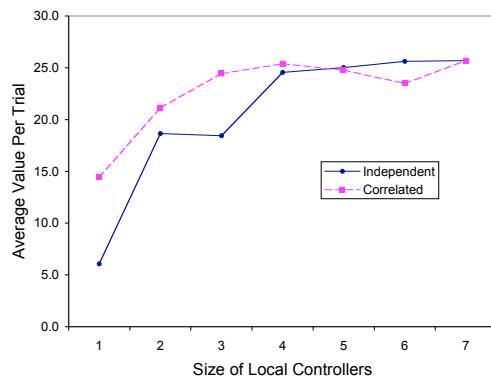
Experiments

- Designed to test how controller size and degree of correlation affect final value
- Experimental domains:
 - **Recycling robot problem** (2 agents, 4 states, 3 actions, 2 observations) [Sutton & Barto 98](#)
 - **Multi-access broadcast channel** (2 agents, 4 states, 2 actions, 6 observations) [Ooi & Wornell 96](#)
 - **Meeting on a grid** (2 agents, 16 states, 5 actions, 4 observations)

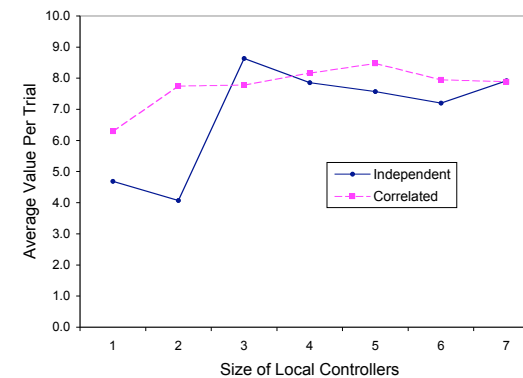
Experimental Setup

- Varied local controller size from 1 to 7 nodes and correlation device size from 1 to 2 nodes
- For each size, we performed 20 trial runs:
 - Initialize action selection and transition functions to be deterministic, with outcomes drawn uniformly
 - Perform 50 backups using randomly selected nodes (values usually stabilized after this point)
- Recorded value from a start distribution

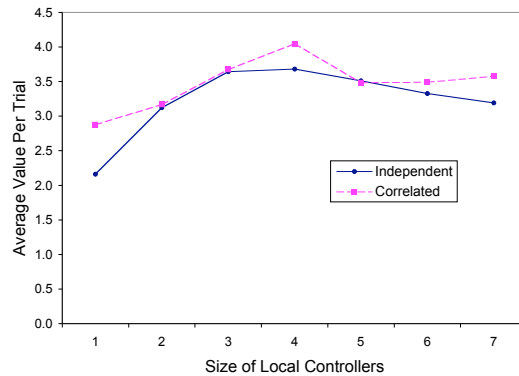
Recycling Robots



Broadcast Channel



Meeting on a Grid



Results

- Correlation leads to higher values on average
- Larger local controllers tend to yield higher average values *up to a point*
 - Problems with improving one controller at a time
 - As controllers grow, it may become easier to get stuck in a local optimum

Conclusion

- BPI has many desirable theoretical properties: **correlated policies**, **fixed memory**, **polynomial time complexity**, **monotonic value improvement**
- Experimental results show a tradeoff between computational complexity and solution quality (more to be explored here)

Future Work

- Implementing bounded PI:
 - Principled order for updating nodes
 - Techniques for escaping local optima
 - Extending bounded PI:
 - Adversarial situations
 - Large numbers of weakly-interacting agents
- Nair et al. 05